

Inferring Task Goals and Constraints using Bayesian Nonparametric Inverse Reinforcement Learning

Daehyung Park

daehyung@csail.mit.edu

Michael Noseworthy

mnosew@mit.edu

Rohan Paul

rohanp@csail.mit.edu

Subhro Roy

subhro@csail.mit.edu

Nicholas Roy

nickroy@csail.mit.edu

Computer Science and Artificial Intelligence Laboratory

Massachusetts Institute of Technology

Cambridge, MA 02139

Abstract: Recovering an unknown reward function for complex manipulation tasks is the fundamental problem of *Inverse Reinforcement Learning* (IRL). Often, the recovered reward function fails to explicitly capture implicit constraints (e.g., axis alignment, force, or relative alignment) between the manipulator, the objects of interaction, and other entities in the workspace. The standard IRL approaches do not model the presence of locally-consistent constraints that may be active only in a section of a demonstration. This work introduces *Constraint-based Bayesian Nonparametric Inverse Reinforcement Learning* (CBN-IRL) that models the observed behaviour as a sequence of subtasks, each consisting of a goal and a set of locally-active constraints. CBN-IRL infers locally-active constraints given a single demonstration by identifying potential constraints and their activation space. Further, the nonparametric prior over subgoals constituting the task allows the model to adapt with the complexity of the demonstration. The inferred set of goals and constraints are then used to recover a control policy via constrained optimization. We evaluate the proposed model in simulated navigation and manipulation domains. CBN-IRL efficiently learns a compact representation for complex tasks that allows generalization in novel environments, outperforming state-of-the-art IRL methods. Finally, we demonstrate the model on two tool-manipulation tasks using a UR5 manipulator and show generalization to novel test scenarios.

Keywords: Learning from Demonstration, Inverse Reinforcement Learning, Tool-use and Manipulation

1 Introduction

Consider the problem of learning a manipulation task from a single demonstration (e.g., see Fig. 1). Inverse reinforcement learning (IRL) [1, 2, 3, 4] has been demonstrated as an effective technique for allowing a robot to learn a task from demonstrations. By representing a task as given by a reward function, a function of state features, IRL has shown how a set of demonstrations can generalize across multiple environments or variations of the task. However, despite many successes [2, 4, 5], conventional IRL has also been shown to be data-intensive [6, 7, 8] and has difficulty with explicitly capturing local properties like alignment or contact constraints necessary for learning the task [9, 10, 11]. *Bayesian Nonparametric IRL* (BN-IRL) [12] has provided a path forward towards learning with considerably smaller number of demonstrations by decomposing a demonstrated task into a sequence of simpler subtasks, providing superior performance, particularly in large state spaces.

Certain types of robot manipulation tasks represent a new challenge for learning from demonstration, in that these tasks require planning with complex constraints such as changing alignment, forces, relative pose etc. Incorporating such abstract constraints [13, 14, 15] is especially important when generalizing demonstrations of seemingly simple tasks such as assembly or removal to the same

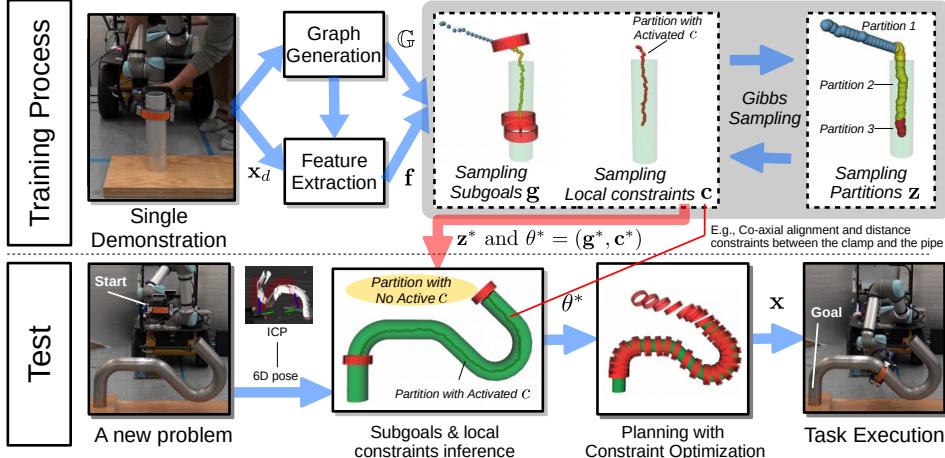


Figure 1: Training and test procedures of CBN-IRL for a clamp-passing task. Given a single demonstration collected from a simple *straight-pipe* environment, CBN-IRL learns subgoals g and local constraints c , such as co-axial alignment and distance constraints between the clamp and the pipe, via blocked Gibbs sampling. In testing, the inferred g and c on feature space are used to plan a locally-optimized trajectory in novel environment (i.e., *multibend-pipe* environment).

task for more complex objects. Conventional reward function parameterizations [9, 10, 11, 5] do not explicitly model constraints, especially when the constraints are non-linear or are represented by low-dimensional manifolds in the state space. The problem of encoding a task entirely in the reward function is exacerbated when learning tasks from demonstration, as different constraints may be active at different parts of the demonstrated trajectory, and the activation of specific constraints may not be fully known or observed [16, 17, 18]. BN-IRL’s decomposition into more subtasks allow to better represent demonstrations with constraints, but will increase the overfitting.

A number of constraint learning approaches [19, 20, 21, 22, 23, 24, 25] determine constraints that *globally* characterize the overall manipulation task. In particular, Englert et al. [26] and Armesto et al. [21] infer overall *task-space constraints* for actions such as pushing and door opening in an inverse optimal control framework. Zhou and Li [23] extract globally valid *safety* constraints by interleaving policy learning and logical model checking. Similar to our work, Chou et al. [27] also recover globally valid constraints on feature space. Ureche et al. [19] postulate force and interaction constraints and learn constraint-parameterizations from demonstration data. Note that these approaches do not consider the presence of local constraints that often exist in complex demonstrations. This paper is closely related to approaches that model both the task partitions and locally-active constraints. For example, Li and Berenson [16] analyze the geometry around the demonstrated trajectory through sampling in task space, and then determine the constraints by segmenting and analyzing the feasible samples. Similarly, Konidaris et al. [18] perform demonstration segmentation using a change-point detection algorithm followed by learning a local reward function each partition. However, these approaches treat the subgoal prediction and constraint learning as separate estimation tasks.

In this work, we propose *Constraint-Based BN-IRL* (CBN-IRL) that models a complex demonstration as a collection of subtasks where each subtask is characterized by a subgoal and a set of local constraints. Learning from a single demonstration, CBN-IRL can infer the set of local constraints from state features and regions where the constraints are likely to be active. Without loss of generality, modelling local constraints allows more complex behaviour to be compactly encoded in different parts of the state space and reduces the number of subtasks required to model the entire demonstration relative to previous results. We also provide an efficient inference algorithm to extend BN-IRL’s Gibbs sampling algorithm to additionally infer local constraints and state space regions where they are applicable. The nonparametric prior over subtasks and constraints allows the model to expand with the complexity of the demonstration. We demonstrate how CBN-IRL can be applied to high-dimensional continuous manipulation tasks and evaluate its performance against other IRL algorithms in simulated 2D navigation and 5D manipulation domains. We show that CBN-IRL can efficiently encode and decode locally constrained behaviors that the other baseline approaches cannot reproduce. Finally, we demonstrate CBN-IRL with two tool-manipulation tasks using a UR5 manipulator.

2 Background: Bayesian Nonparametric IRL (BN-IRL)

The section presents an overview of the BN-IRL formulation that forms the basis of the model introduced in this work. For a detailed exposition, we refer the reader to [12].

Conventional IRL aims to recover the reward function for a *Markov Decision Process* (MDP) defined by the 5-tuple, (S, A, T, R, γ) . Here, the symbols S , A , and $\gamma \leq 1$ denote the state space, the action space, and the discount factor, respectively. $T : S \times A \rightarrow Pr(S)$ is a stochastic transition function and $R : S \times A \rightarrow \mathbb{R}$ is a reward function. The robot observes a set of demonstrations represented as a time-ordered set of observed unique state-action pairs $\mathcal{O} = \{\mathcal{O}_1, \mathcal{O}_2, \dots\}$, where an observation $\mathcal{O}_i = (s_i \in S, a_i \in A)$. *Bayesian Nonparametric IRL* (BN-IRL) models the observed demonstration as an MDP but postulates partitioning of the observations into smaller groups and learning a set of simple reward functions modeling local properties of each group. Formally, each partition is characterized with a subgoal g and a reward function $R_g(s_i) = \mathbb{1}(s_i = g)$. For each time step in the demonstration, a latent assignment-variable $z_i \in \mathbb{N}$ is introduced to associate the observation at each time step i with a specific partition of the observed data and corresponding subgoal. The distribution over assignments from an observation \mathcal{O}_i to partition z_i is modeled by a DP prior¹ to represent a priori unknown number of subgoals.

BN-IRL infers the most likely set of partitions, the corresponding subgoals, and the assignment-variables given demonstrations. The joint distribution over the set of observations \mathcal{O} , the goals \mathbf{g} , and the partition variables \mathbf{z} is expressed as:

$$p(\mathcal{O}, \mathbf{g}, \mathbf{z}) = \prod_{i=1}^N \underbrace{p(\mathcal{O}_i | g_{z_i})}_{\text{Obs. Likelihood}} \underbrace{p(z_i | z_{-i}, \eta)}_{\text{DP Partition Prior}} \prod_{j=1}^{J_N} \underbrace{p(g_j)}_{\text{Goal Prior}}. \quad (1)$$

Here, g_{z_i} , $N \in \mathbb{N}$, $J_N \in \mathbb{N}$, and $\eta \in \mathbb{R}$ are the goal assigned in a partition z_i , the number of observations, the number of partitions after obtaining N observations, and the concentration constant of DP, respectively. The conditional distribution over the assignment-variables is represented as:

$$p(z_i | z_{-i}, \eta) = \begin{cases} \frac{m_i}{n-1+\eta} & \text{if } 1 \leq i \leq K \\ \frac{\eta}{n-1+\eta} & \text{Otherwise,} \end{cases}$$

where n , m_i , and K are the number of observations, observations in partition i , and partitions, respectively. The nonparametric DP prior with concentration parameter η characterizes the demonstration by a potentially infinite set of partitions enabling the model to expand appropriately with new observations. The BN-IRL model infers the posterior distribution over the subgoals and partition assignments, (\mathbf{g}, \mathbf{z}) , given the demonstration. Subsequently, the *maximum a posteriori* (MAP) subgoals for each partition and assignment variables associated with the demonstration are determined as: $(\mathbf{g}^*, \mathbf{z}^*) = \arg \max_{\mathbf{g}, \mathbf{z}} p(\mathbf{g}, \mathbf{z} | \mathcal{O})$. The inference over (\mathbf{g}, \mathbf{z}) is performed using uncollapsed Gibbs sampling [29], where $z_i \in S$ and $1 \leq k \leq |S|$.

3 Constraint-based Bayesian Nonparametric IRL (CBN-IRL)

We present *Constraint-based BN-IRL* (CBN-IRL), a novel extension for BN-IRL, that posits the presence of local constraints characterizing subgoals constituting a single observed demonstration.

Formally, the sub-demonstration in the i th partition is viewed as a sample from the optimal policy for an MDP given subgoal and a subset of active constraints, $c_i : c_i(s) \rightarrow \{1, 0\}$ as well as a subgoal, g_i . Each constraint in an inferred partition places strict restrictions on which states an agent can visit when the constraint is active. Hence, we define an $MDP/(R, T) + (R_g, T_c)$ that is an MDP for which R and T are not specified, but IRL provides a subgoal-based reward R_g and a constraint-based transition function T_c . The problem therefore is to infer the decomposition of the demonstrated trajectory into smaller partitions and for each partition and estimating the subgoal and associated constraints that maximize the likelihood of the observed trajectory (i.e. the MDP policy). Note that CBN-IRL requires a single demonstration that is a set of unique state-action pairs, referred to as observations \mathcal{O} .

Discretized State-Action Space: The CBN-IRL assumes an MDP with a discrete state space. However, for high-dimensional manipulation problems, a regular discretization of the configuration space is intractable to maintain, hence we use a sampling-based strategy for state-space discretization.

¹BN-IRL uses the *Chinese Restaurant Process* model of the Dirichlet Process [28].

Conventional sampling methods often require a large number of samples to obtain valid states in narrow passages [30]. We address this issue by combining the sampling and expansion strategy of the *probabilistic roadmap* (PRM) [31] and the *rapidly-exploring random trees* (RRT) [32], similar to the multi-component rapidly-exploring roadmap (MC-RRM) [20]. We first generate an initial state-action graph with node set V and edge set E , $\mathbb{G} = (V, E)$, from a guidance path \mathbf{x}_g that is a linearly interpolated state path from a start to a goal including any potential waypoints². Note that the waypoints are not required, but fewer waypoints often require longer sampling time to obtain a well-connected graph. The initial graph consists of multiple disconnected components $\mathbf{V} = \{V_1, V_2, \dots\}$ due to the maximum length of edges d_{\max} . We then sample each state s_{sample} from a Gaussian distribution over \mathbf{x}_g . The dense sampling near the boundary of objects helps to keep connectivity in narrow regions with poor visibility [33]. Otherwise, we need a large amount of samples to obtain a well-connected graph. We then incrementally expand each component V_i by adding at most $d_{\max} \in \mathbb{R}$ distanced state s_{new} from a near state s_{near} toward V_i and s_{sample} until all components are connected $|\mathbf{V}| = 1$ with the desired number of states. We neither eliminate all states in collision nor zero the transition probability assuming the collision constraint that has to be learned or observed. Instead, to expedite the learning process in this paper, we sample a part of collision states (i.e., boundary states) and collision-free states (see Fig. 2). As samples increase, we get a connected graph from a start to a goal including a variety of paths. As with PRM, our method samples a roadmap that enables to optimize state paths as the number of states increases and the RRT guarantees expansion of the roadmap to valid states in narrow state passages.

Learning the subgoals and constraints of this MDP requires the demonstrated trajectory to be projected onto the state space keeping the topological connectivity. This is accomplished via a modification to the Dijkstra's algorithm similar to [34], we can find the shortest path but close to a demonstrated trajectory on \mathbb{G} . We use a weighted transition cost as the current node s_c 's path cost, $\min_{\forall s_k \in \mathbb{E}(s_c)} \gamma d(s_c, s_k) + (1 - \gamma)d(s_k, s_t)$, where \mathbb{E} returns a list of nodes connected on s_c 's edges, γ is a weight ($0 \leq \gamma \leq 1$), and $d(\cdot, \cdot)$ is a distance metric between two states. For our 5D tool manipulation tasks in Sec. 4, we use a weighted sum of Euclidean position distance and angular difference between two input poses. s_t is a reference state on the demonstrated trajectory. We use grid search to set s_t in the algorithm to precisely project the demonstration.

Subgoals and Constraints: We now formalize the notion of subgoals and constraints estimated by the CBN-IRL framework. We use a set of predefined generic spatial features ($\mathbf{f} \in \mathbb{R}^M$) for representing the world state, assuming the robot knows the geometry of environment. In particular, we only incorporate coordinate-free features such as *minimum distance from obstacles*. A subgoal can be a set of features that can be generalized to new environment. Note that, as in [12], we also limit the possible subgoals mapped from features by selecting the nearest ergodic state satisfying constraints. Each constraint operates on the feature space to denote valid and invalid regions of the state space. In details, we define a feature-wise simple-constraint, c^i , which is computed from the i^{th} feature and used to check if a feature, $f^i \in \mathbf{f}$ falls within a bound $\zeta \in \mathbb{R}$ of an expected feature value \hat{f}^i : $c^i : |f^i - \hat{f}^i| - \zeta \leq 0 \rightarrow \{1, 0\}$. In practice, we estimate \hat{f}^i as the average of observed features associated with the same partition. The free parameter ζ is determined empirically in order to maximize the similarities from demonstrations and can also be used to control the tightness of constraints. A constraint can be viewed as a Boolean function of defined over the feature space. Formally, a constraint c is a conjunction of active simple-constraints at a given partition such that all simple-constraints are satisfied: $c = c^1 c^2 \dots c^K$. K denotes the number of features used to define c .

CBN-IRL induces an MDP(R_g, T_c) with an unknown reward function R_g for subgoal g and a constraint-dependent transition T_c for an active constraint c . The presence of active constraints disallows transitions into state space regions that violate the indicated constraints. Hence, the constraint-dependent *truncated* stochastic transition function T_c can be constructed as:

$$T_c(s, a, s') = \begin{cases} 0 & \text{if } c(s') = \text{False} \\ T(s, a, s') / \sum_{s''} T(s, a, s'') & \text{Otherwise,} \end{cases} \quad (2)$$

²For the pipe environments in Sec. 4, we linearly interpolated a start, any states on the edge or inflection point on each pipe, and a goal to generate a guidance path \mathbf{x}_g , where we set $|\mathbf{x}_g| = |\mathcal{O}|$.

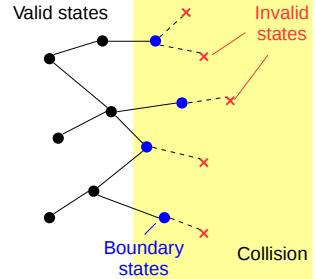


Figure 2: Collision-free and boundary states for graph generation

where $T(s, a, s')$ is a transition probability function with $\sum_{s'} T(s, a, s') = 1$, and $s'' \in \{s | s \in S, c(s) = \text{True}\}$ denotes the subset of states that satisfy the constraint.

Inference: We now detail the process of trajectory generation as solving a sequence of MDPs given the same state-action spaces but different subgoals. Note that the subgoals and constraints are latent variables $\theta = \{(g_1, c_1), (g_2, c_2), \dots\}$ in the model. We assume that the subgoals g are a subset of the states observed in an expert demonstration and that the expert demonstration satisfies all task constraints. The joint distribution of a new state-action trajectory of length l , $\mathcal{O}_{\text{new}} = \{(s_0, a_0), (s_1, a_1), \dots, (s_l, a_l)\}$, and a demonstration, \mathcal{O} , can be represented as:

$$p(\mathcal{O}_{\text{new}}, \mathcal{O}) = p(\mathcal{O}_{\text{new}} | \mathcal{O}) p(\mathcal{O}) = \int_{\theta, \mathbf{z}} \underbrace{p(\mathcal{O}_{\text{new}} | \theta, \mathbf{z})}_{\text{Likelihood}} \underbrace{p(\theta, \mathbf{z} | \mathcal{O})}_{\text{Subgoals \& Constraints Likelihood}} \underbrace{p(\mathcal{O})}_{\text{Prior}}. \quad (3)$$

The set of latent variables (θ, \mathbf{z}) are inferred from a demonstration by inferring the MAP estimate of the likelihood of subgoals, active constraints for each partition, and assignment variables associated with the demonstration: $(\theta^*, \mathbf{z}^*) = \arg \max_{\theta, \mathbf{z}} p(\theta, \mathbf{z} | \mathcal{O})$. Direct inference is intractable since the joint space of latent discrete partition assignment variables and the set of constraints (θ, \mathbf{z}) is combinatorially large and also greater than that BN-IRL due to additional inference over constraint variables. Building on [12], we adopt approximate MCMC inference using Blocked Gibbs Sampling, which iteratively samples (\mathbf{g}, \mathbf{c}) and \mathbf{z} by conditioning on the other observed variables. After running Gibbs sampling, the empirical mode of the samples $\theta_{1:t}$ and $\mathbf{z}_{1:t}$ will have converged to values that approximate the true values of the latent parameters as the number of sampling iterations $t \rightarrow \infty$ and the Markov chain producing samples is ergodic according to Theorem 1 in [12]. We describe the sampling updates of \mathbf{z} and θ below.

Partition Likelihood: The conditional probability distribution of a partition z_i given the other partitions, z_{-i} , observations, \mathcal{O} , and goals and constraints, θ , can be expressed as:

$$p(z_i | z_{-i}, \theta, \mathcal{O}, \eta) \propto \underbrace{p(\mathcal{O}_{z_i} | \theta_{z_i})}_{\text{Likelihood}} \underbrace{p(z_i | z_{-i}, \eta)}_{\text{DP Prior}}, \quad (4)$$

where \mathcal{O}_{z_i} are the observations assigned to z_i and depend on the subgoal and local-constraint pair θ_{z_i} for that partition. The likelihood represents the likelihood that an agent acting optimally under the current constraints and seeking the estimated subgoal generates the observations as:

$$p(\mathcal{O}_{z_i} | \theta_{z_i}) = p(a_i | s_i, g_{z_i}, c_{z_i}) = \frac{e^{\alpha Q^*(s_i, a_i, R_{g_{z_i}}, c_{z_i})}}{\sum_j e^{\alpha Q^*(s_i, a_j, R_{g_{z_i}}, c_{z_i})}}. \quad (5)$$

Here, hyperparameter α represents the degree of confidence in the expert's ability to maximize reward and Q^* denotes the optimal Q -function for the MDP with reward function R_g and the constraint-conditioned truncated transition function T_c .

Subgoals and Constraints Likelihood: The conditional likelihood of a subgoal with the associated local-constraint, $\theta_j = (g_j, c_j)$, given a specific partition \mathbf{z} can be expressed as:

$$p(\theta_j | \mathbf{z}, \mathcal{O}) \propto p(\mathcal{O}_j | \theta_j, \mathbf{z}, \mathcal{O}_{-j}) p(\theta_j | \mathbf{z}, \mathcal{O}_{-j}) = \sum_{i \in I_j} \underbrace{p(\mathcal{O}_i | \theta_{z_i})}_{\text{Likelihood}} \underbrace{p(\theta_j)}_{\text{Prior}}, \quad (6)$$

where $I_j = \{i : z_i = j\}$. Joint inference over subgoals and local constraints is computationally expensive due to the combinatorially large space of constraints since a constraint is a conjunction of any subset of potential simple-constraints. However, the inference complexities are not directly comparable with that of BN-IRL due to the convergence of Gibbs sampling. We also reduce the complexities by sampling subgoals from the set of observations assigned to the given partition. We exclude simple-constraints defined by a feature that is not consistent at any given time window. The coordinate-free features (e.g., *distance from objects*) allows constraints that can generalize to new goals. We then limit the class of constraints considered to either be a conjunction of all simple-constraints or the empty set representing no constraints: $c_j = \prod_{\{k | f^k \in \mathbf{f}_c\}} c_j^k$ or \emptyset .

The likelihood is calculated as in the previous section. The runtime efficiency of model inference is improved by pre-computing the Q^* values for each subgoal and constraint combination (g_i, c_i) .

As the sampling process requires re-estimation of the valid feature range, \hat{f}_j^k , used to compute constraints based on partition assignments at each update, it is hard to pre-compute Q^* as previously done. To resolve these issues, we pre-compute a list of potential expected feature values, $F = \{\hat{f}^{k,1}, \hat{f}^{k,2}, \dots, \hat{f}^{k,M}\}$, from a discretized feature space, where M is the resolution and the extent of the space $(\hat{f}^{k,1}, \hat{f}^{k,M})$ is obtained from the minimum and maximum value of features in the demonstration \mathcal{O} , respectively. CBN-IRL then pre-computes Q per potential constraint defined from each feature-mean in F . During the sampling, CBN-IRL selects a pre-computed Q from the closest pre-computed feature-mean as Q^* . CBN-IRL finally predicts a current partition z of a current state s and then predicts the maximum likelihood of action a given the current z and the empirical mode of samples (θ^*, \mathbf{z}^*) : $z^*, a^* = \arg \max_{z_i, a} p(a|s, g_{z_i}, c_{z_i})$. In practice, we hold z_i until reaching g_{z_i} .

Trajectory Planning for Robot Manipulation: Finally, we address the problem of generating a continuous trajectory to attain the set of inferred subgoals constituting the overall task while respecting the inferred local constraints. Formally, we seek a smooth trajectory $\mathbf{x} = \{s_1, \dots, s_l\}$ that minimizes the sum of displacements, while satisfying a set of constraints. At j th partition, we formulate a constrained optimization problem, $\min_{\mathbf{x}=\{s_1, \dots, s_l\}} \sum_{t=1}^{l-1} \|s_{t+1} - s_t\|^2$ subject to $s_1 = \text{initial state}$, $s_l = g_i$, and $\forall k \quad c_j^k(s) = |f^k - \hat{f}_j^k| - \zeta \leq 0$. In this work, we solve the problem by using an optimization-based planner, *TrajOpt* [35].

4 Experimental Setup

2D Navigation Tasks: As a preliminary test, we evaluated CBN-IRL and four baseline methods, such as MaxEnt [4] and Deep-MaxEnt IRL [6], with respect to their ability to reproduce the demonstrated trajectory both within the same simulated environment and in novel environments. We developed this testbed as an environment of OpenAI Gym [36], a reinforcement learning toolkit. Fig. 3 shows the first environment requires the agent to move from a start location to a goal location without colliding with the wall (black) while keeping a certain distance from the wall. For training, we collected an expert trajectory that satisfies all task constraints and projected it onto a uniformly-randomly sampled graph ($\|V\| = 5,000$). In testing, each method then produced a trajectory on the graph. Performance was measured by the loss (i.e., *Dynamic Time Warping* (DTW) in [37]) between generated and demonstrated trajectories.

We also evaluated generalization performance by comparing CBN-IRL with BN-IRL in 1,000 randomized sinusoidal passages ($= h \cdot \sin(w\theta + \theta_0)$) encompassed by left/right walls, uniformly-randomly sampled (h, w, θ_0) in $0.5 \leq h \leq 3$, $0.5 \leq w \leq 3$, and $-\pi \leq \theta_0 \leq \pi$. Each method is trained on a single demonstrated passage. We generated a path that keeps a certain distance between two passage walls to measure performance against. We used as many features as possible: distance ratio between two walls and distances from start, goal, and four edges of left/right walls ($\in \mathbb{R}^7$).

5D Tool Manipulation Tasks: Next, we evaluated our framework with simulated and real-world scenarios in continuous large-state space using a UR5 manipulator. We formulated two tool-manipulation tasks, screwdriver insertion and clamp-passing tasks, by compactly sampling discretized state-action space and learning subgoals and local constraints from an expert demonstration. The following state-space features were used in the experiment:

- *Progress*: Path length from the pipe opening to the clamp state s scaled by the total length ($\in \mathbb{R}$),
- *Distance from Pipe*: Distance between the clamp state s and the nearest pipe center s_{pipe} normalized by the sum of distances from the start s_0 and s_{pipe} ($= d(s, s_{\text{pipe}})/(d(s_t, s_0) + d(s_t, s_{\text{pipe}}))$).
- *Center Alignment*: Euclidean position distance from a state to the pipe centerline ($\in \mathbb{R}$),
- *Relative Orientation*: Angular displacement between the clamp and the pipe orientation ($\in \mathbb{R}$),
- *Collision*: A binary feature indicating collision between the clamp and the pipe ($\in \{0, 1\}$).

The estimated goals and constraints are used in a constrained optimization *TrajOpt* [35] to determine the low-level continuous trajectory of the robot. The robot’s perception system included a wrist-mounted Intel RGB-D camera. The 6DOF pose of the *multibend*-pipe was determined using *Iterative Closest Point* (ICP) between the depth point cloud and the prior geometric model of the object.

5 Evaluation

2D Navigation Tasks: We show the benefit of modelling constraints by learning to reproduce a non-linear trajectory in a 2D semi-circular obstacle environment, as shown in Fig. 3 **Top**. BN-IRL

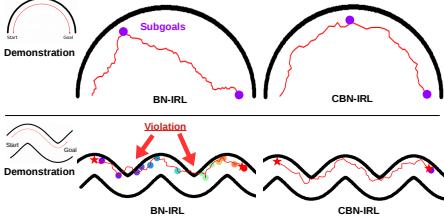


Figure 3: **Top:** CBN-IRL produces a demonstration-like semi-circular path that BN-IRL with two subgoals cannot imitate. **Bottom:** BN-IRL with many subgoals cannot produce a valid path on a complex environment, but CBN-IRL can produce it using a goal with constraints.

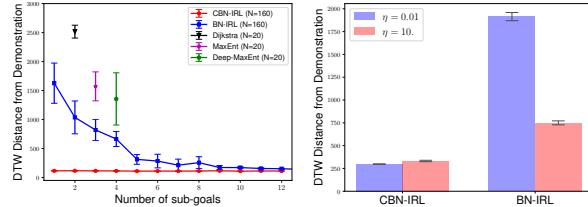


Figure 4: **Left:** Comparison with four baseline given a semi-circular demonstration where training and testing environments are identical. **Right:** Generalization performance in 1,000 randomized sinusoidal path environment. CBN-IRL results in consistently small DTW distances from demonstrations regardless of the number of subgoals determined by η in DP.

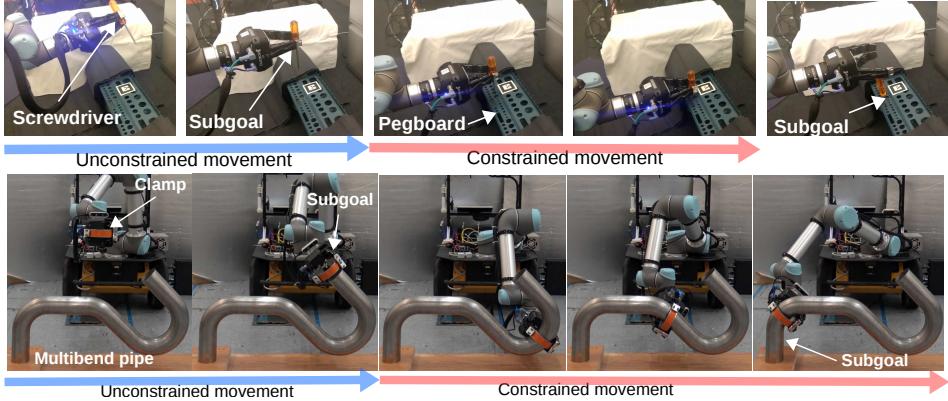


Figure 5: Screwdriver insertion and clamp-passing experiments. A UR5 robot successfully placed tools by selectively activating the inferred subgoals and local constraints in each novel environment that requires strict co-axial orientation constraints around the space of target objects.

inferred two subgoals but failed to keep a consistent distance from the obstacle. On the other hand, CBN-IRL also found two subgoals but was able to produce an expert-like trajectory by limiting its actions to stay close to the obstacle. Note that we lowered the tightness of constraints due to the coarse resolution of discrete graph. Fig. 4 **Left** shows CBN-IRL always resulted in the lowest dissimilarity (i.e., DTW distance) from the demonstration regardless of the number of subgoals. By increasing the number of subgoals (by increasing a parameter η), BN-IRL was able to reduce the dissimilarity from the expert demonstration. Note that we set training and testing environments are identical to show the representation power of constraints. This indicates the constraints are able to capture a type of property in demonstrations that cannot be easily represented by reward functions. Conventional IRL methods resulted in significantly lower performance due to the limitation of reward function or the single demonstration. Note that these baselines used the same features as in CBN-IRL.

Next, we show the generalization performance by comparing with BN-IRL in randomized sinusoidal passages, as shown in Fig. 3 **Bottom**. An agent is given a single demonstration of a passage, and then is given a new shape and its initial state and told to repeat the task. Fig. 4 **Right** shows CBN-IRL resulted in lower average dissimilarity than BN-IRL. Regardless of the number of subgoals, CBN-IRL was able to produce consistent results using the learned constraints, whereas BN-IRL resulted in the higher dissimilarity given $\eta = 0.01$ and 10 , which recovered 2 and 16 subgoals, respectively. The simple reward structure from BN-IRL with few subgoals is not enough to draw sinusoidal curves, though more subgoals decrease the dissimilarity. This emphasizes the fact that the subgoal-only model requires significantly more subgoals to reproduce complex demonstration.

5D Tool Manipulation Tasks: Next, we evaluate the generalization performance of our framework in more realistic scenarios toward manipulation tasks in high-dimensional continuous space. Table 1 shows the comparison between CBN-IRL and other four baseline methods in clamp-passing tasks. We trained each method given a single clamp-passing demonstration in a *straight-pipe* environment and then tested each to reproduce a demonstration-like trajectory given uniform-randomly sampled

Table 1: Comparison of generalization performance in 5D clamp-passing tasks. We trained each method on a single demonstration of the *straight*-pipe environment and then tested the performance on novel environments. The numbers represent the number of successful completions given uniform-randomly sampled 100 initial poses. Each method with trajectory optimization (TO) shows completion performance in continuous space. The two pairs of results show when we increase the number of subgoals by setting $\eta = 10^{-3}$ and $\eta = 10$, where we inferred 3 and 17 subgoals respectively for BN-IRL and 2 and 8 subgoals for CBN-IRL.

Env. Method \ Env.	Training & Test		Test	
				
TO + Human	100/100	100/100	16/100	0/100
TO + MaxEnt [4]	99/100	100/100	39/100	0/100
TO + BN-IRL [12]	100/100	100/100	0/100	0/100
TO + CBN-IRL	100/100	100/100	97/100	95/100
		100/100	100/100	98/100

100 initial clamp poses in four novel pipe environments: *straight*, 90° , 180° , and *multibend* pipes. We produced an optimized continuous trajectory given the learned/designed reward functions and constraints. For example, *TO+Human* and *TO+MaxEnt* are trajectory optimizations with manually-designed and learned reward functions, respectively. We also attempted to implement a conventional planning algorithm (i.e., Dijkstra algorithm) as a baseline method. The algorithm succeeded where a single subgoal could be used and constraints (i.e., collision avoidance) were manually specified, otherwise it did not succeed.

Our *TO+CBN-IRL* consistently demonstrates the highest success rates in all environments whereas *TO+BN-IRL* failed at all attempts to make the clamp pass through new complex pipes though we increased the number of subgoals from 2 to 16 by setting $\eta = 10$. That is because having a large number of subgoals does not guarantee a high-resolution model of non-linear curve to pass the various length of pipes. In other words, many subgoals can be useful to describe a specific problem, but hard to be generalized to other problems. Manually-specified or learned MaxEnt [4] reward functions also showed limitations to solve complex passing problems with 180° and *multibend* pipes. This statistical comparison shows learning constraints compactly and efficiently models demonstrations while generalizing to novel environments.

We finally demonstrated the robustness of CBN-IRL on a real robot, UR5. Fig. 5 shows two representative tool manipulation experiments³. CBN-IRL learned two subgoals and local constraints from an expert demonstration on simpler simulated environments: two-hole pegboard and *straight* pipe. In the first insertion task, by inferring subgoals and local constraints in new environments, the robot successfully inserted a screwdriver into the hole while keeping the relative orientation after reaching the first subgoal on top of the pegboard. Likewise, for the clamp-passing task that requires strict co-axial constraints between the clamp and the pipe center, the robot activated the constraints after reaching the entrance of the pipe (i.e., a subgoal) and successfully passed the grasped clamp through the complex *multibend* curves without collision. The experiment demonstrated that CBN-IRL can precisely recover constraints to generalize the learned skills in novel manipulation scenarios.

6 Conclusion

We introduced *Constraint-based Bayesian Nonparametric Inverse Reinforcement Learning* (CBN-IRL) that learns local constraints and their activation space from a single demonstration. CBN-IRL partitions a demonstration into a series of subtasks each represented by a subgoal and set of local constraints. Our approach increases the expressiveness while maintaining the data efficiency of BN-IRL. Our method does not require the number of subgoals and constraints to be pre-specified, and the addition of constraints results in a more compact representation and therefore computationally efficient approach overall. By adding constraints, our method was able to significantly reduce the divergence from a demonstration and generalize to new environments without increasing the number of subgoals. We also showed that CBN-IRL can be extended to high-dimensional tool-manipulation tasks by projecting the demonstration on discrete space and producing a continuous trajectory via constrained optimization. Our demonstration showed that CBN-IRL successfully completed the assigned manipulation tasks.

³A video for the demonstration is submitted as a multimedia extension along with this manuscript.

Acknowledgments

We gratefully acknowledge funding support in part by the U.S. Army Research Laboratory under the Robotics Collaborative Technology Alliance (RCTA) Program, Lockheed Martin Co., the Toyota Research Institute Award LP-C000765-SR, and Honda Research Institute.

References

- [1] A. Y. Ng, S. J. Russell, et al. Algorithms for inverse reinforcement learning. In *Proc. Int'l Conf. on Machine Learning*, pages 663–670, 2000.
- [2] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proc. Int'l Conf. on Machine Learning*, page 1. ACM, 2004.
- [3] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.
- [4] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *Proc. Nat'l Conf. on Artificial Intelligence*, 2008.
- [5] C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *Proc. Int'l Conf. on Machine Learning*, pages 49–58, 2016.
- [6] M. Wulfmeier, P. Ondruska, and I. Posner. Maximum entropy deep inverse reinforcement learning. In *Adv. Neural Information Processing Systems - Deep Reinforcement Learning Workshop*, 2015.
- [7] S. Gu, E. Holly, T. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *Proc. Int'l Conf. on Robotics and Automation*, pages 3389–3396. IEEE, 2017.
- [8] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *Proc. Robotics: Science and Systems*, 2018.
- [9] A. Boulieras, J. Kober, and J. Peters. Relative entropy inverse reinforcement learning. In *Proc. Int'l Conf. on Artificial Intelligence and Statistics*, pages 182–189, 2011.
- [10] A. Doerr, N. D. Ratliff, J. Bohg, M. Toussaint, and S. Schaal. Direct loss minimization inverse optimal control. In *Proc. Robotics: Science and Systems*, 2015.
- [11] J. Ho and S. Ermon. Generative adversarial imitation learning. In *Adv. Neural Information Processing Systems*, pages 4565–4573, 2016.
- [12] B. Michini and J. P. How. Bayesian nonparametric inverse reinforcement learning. In *Proc. Joint European Conf. on Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer, 2012.
- [13] M. Toussaint, K. R. Allen, K. A. Smith, and J. B. Tenenbaum. Differentiable physics and stable modes for tool-use and manipulation planning. In *Proc. Robotics: Science and Systems*, 2018.
- [14] L. V. Nair, N. S. Srikanth, Z. Erikson, and S. Chernova. Autonomous tool construction using part shape and attachment prediction. In *Proc. Robotics: Science and Systems*, June 2019.
- [15] A. Xie, F. Ebert, S. Levine, and C. Finn. Improvisation through physical understanding: Using novel objects as tools with visual foresight. In *Proc. Robotics: Science and Systems*, 2019.
- [16] C. Li and D. Berenson. Learning object orientation constraints and guiding constraints for narrow passages from one demonstration. In *Proc. Int'l Symp. on Experimental Robotics*. Springer, 2016.
- [17] O. Kroemer, C. Daniel, G. Neumann, H. Van Hoof, and J. Peters. Towards learning hierarchical skills for multi-phase manipulation tasks. In *Proc. Int'l Conf. on Robotics and Automation*, pages 1503–1510. IEEE, 2015.

- [18] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto. Robot learning from demonstration by constructing skill trees. *Int'l J. of Robotics Research*, 31(3):360–375, 2012.
- [19] A. L. P. Ureche, K. Umezawa, Y. Nakamura, and A. Billard. Task parameterization using continuous constraints extracted from human demonstrations. *Trans. on Robotics*, 31(6):1458–1471, 2015.
- [20] G. Ye and R. Alterovitz. Demonstration-guided motion planning. In *Int'l Symp. on Robotics Research*, 2011.
- [21] L. Armesto, J. Bosga, V. Ivan, and S. Vijayakumar. Efficient learning of constraints and generic null space policies. In *Proc. Int'l Conf. on Robotics and Automation*. IEEE, 2017.
- [22] M. Toussaint. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *Proc. Int'l Conf. on Artificial Intelligence and Statistics*, pages 1930–1936. AAAI Press, 2015.
- [23] W. Zhou and W. Li. Safety-Aware Apprenticeship Learning. In *Proc. Int'l. Conf. on Computer Aided Verification*, pages 662–680. Springer, 2018.
- [24] A. F. Bobick and A. D. Wilson. A state-based approach to the representation and recognition of gesture. *Trans. on Pattern Analysis and Machine Intelligence*, 19(12):1325–1337, 1997.
- [25] C. Prez-D'Arpino and J. A. Shah. C-LEARN: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy. In *Proc. Int'l Conf. on Robotics and Automation*, pages 4058–4065, May 2017.
- [26] P. Englert, N. A. Vien, and M. Toussaint. Inverse KKT: Learning cost functions of manipulation tasks from demonstrations. *Int'l J. of Robotics Research*, 36(13-14), 2017.
- [27] G. Chou, D. Berenson, and N. Ozay. Learning constraints from demonstrations. In *Proc. Int'l Workshop on Algorithmic Foundations of Robotics*, 2018.
- [28] J. Pitman et al. Combinatorial stochastic processes. Technical report, Dept. Statistics, UC Berkeley, 2002.
- [29] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. In *Readings in Computer Vision*, pages 564–584. Elsevier, 1987.
- [30] D. Hsu, J.-C. Latombe, and H. Kurniawati. On the probabilistic foundations of probabilistic roadmap planning. *Int'l J. of Robotics Research*, 25(7):627–643, 2006.
- [31] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *Trans. on Robotics and Automation*, 12(4), 1996.
- [32] S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report TR 98-11, Computer Science Department, Iowa State University, 1998.
- [33] V. Boor, M. H. Overmars, and A. F. Van Der Stappen. The Gaussian sampling strategy for probabilistic roadmap planners. In *Proc. Int'l Conf. on Robotics and Automation*, 1999.
- [34] A. Byravan, M. Monfort, B. D. Ziebart, B. Boots, and D. Fox. Graph-based inverse optimal control for robot manipulation. In *Proc. Int'l Joint Conf. on Artificial Intelligence*, 2015.
- [35] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel. Finding locally optimal, collision-free trajectories with sequential convex optimization. In *Proc. Robotics: Science and Systems*, volume 9, pages 1–10, 2013.
- [36] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. OpenAI Gym, 2016. URL <https://gym.openai.com/>.
- [37] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *AAAI-94 workshop on knowledge discovery in database*, volume 10, pages 359–370, 1994.